Project Part 2 - Progress Report

Alec Landow and Liam Adams

October 22, 2020

1 Solved Issues

Solved Issue	**Member Responsible**
Development of Skeleton Program	Both
Initial State	Alec Landow
Implementation outline	Liam Adams

We created a skeleton for the program with all of the functions we believe we need to implement. For each function we've either written comments on how to implement it, written pseudocode, or written actual code. We are focusing on implementing the algorithm defined in Mateus and Omar [1].

The first step in this algorithm is to create the initial state of the circuit. This consists of creating a quantum register consisting of $s = lg(N - M + 1)$ qubits to represent the position of each character, where M is the number of characters in the pattern we are searching for within a string of length N. In order to implement this we put the qubits representing the first character of the pattern in a superposition by applying a Hadamard gate to each of them.

Then we entangle each of the s qubits of the first pattern character's index-encoding qubits with the corresponding qubits of the next pattern character's index-encoding group of qubits. Entangling the registers effectively copies the index encoding from one state to the next in preparation for the next step. After entangling, there is another set of CX gates applied that act on the registers that encode the position of the next character in the pattern within the overall string. These CX gates ensure that the indexes of the quantum states representing the pattern characters are consecutive.

The below diagrams display results of our implementation of the initial state for two different inputs.[2][3]

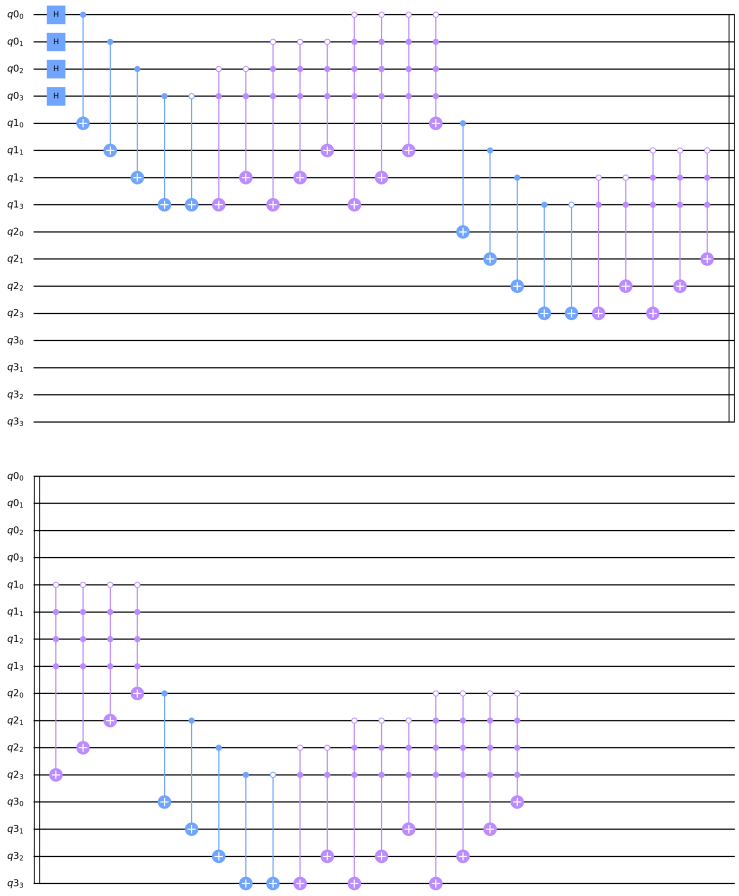

Figure 1: Pattern Size = 3, Input String Size = 11

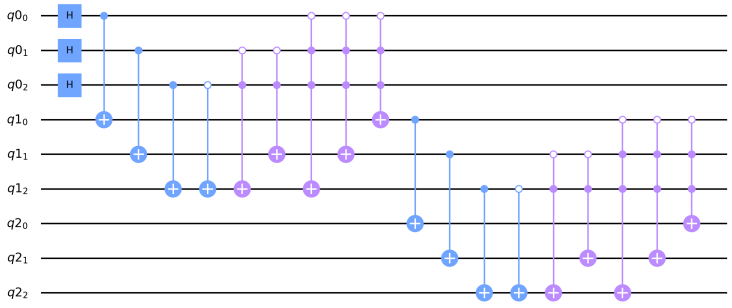

Figure 2: Pattern Size = 2, Input String Size = 9

2 Open Problems

The remainder of the algorithm proceeds as follows. We still need to implement each of the below steps. The overall string is w of length N. The pattern is p of length M.

1. Select a number at random from the range consisting of the length of the string, excluding the pattern plus 1, or $N - M + 1$. Call this number r.

2. Iterate over the range of r, and in each iteration choose an index j at random from [1, M].

   Only need to select $r = N - M + 1$ indices j at random from the pattern.

3. Apply respective query operator Q_j to the current state. The query operator is also known as the oracle function. The function it needs to implement is

$$f(x) = \begin{cases} 1, & \text{if the i}^{th} \text{ letter of string is } \ M_j \\ 0, & \text{otherwise} \end{cases}$$

   If the j^{th} character from p matches a character in w, its state's amplitude will be amplified by the diffusor step below. Since we are selecting indices j at random each character's state amplitude may be amplified more than others.

   The circuit for the oracle function also incorporates Gray codes. A gray code between two binary strings is a sequence of binary strings in which 2 consecutive strings only differ by one bit, eventually the source string evolves to the target string. A transposition gate (labelled CNOT in the paper) is developed to transition strings from string to string (state to state). It is still an open problem to determine the utility of this aspect of the oracle.

   There will be a different query operator Q_j (circuit) for each character in the alphabet.

4. Apply Grover diffusion to current state. This is what amplifies the amplitude of the state.

5. Measure the state. The basis of the state is over the length of w which is N. So the measurement will correspond to an index in the string w which we interpret as the beginning of the closest match of the pattern p.

6. Repeat steps 1-5 $\sqrt{N}$ times

3 Updated Timeline

- Monday, November 5, 2020: Oracles implemented
- Sunday, November 8, 2020: Diffuser implemented
- Sunday, November 17, 2020: Pattern matching program is optimized for use on a real quantum computer

References

[1] P. Mateus and Y. Omar. Quantum pattern matching, 2005.

[2] Is there an anti-control gate in qiskit? https://stackoverflow.com/questions/61286794/is-there-an-anti-control-gate-in-qiskit.

[3] Qiskit 0.23.0 documentation. https://qiskit.org/documentation/index.html.